65

What is claimed is

1   1. A method of test generation for testing computer
2   software, comprising the steps of:
3   modeling a software application as a finite state ma-
4   chine to define a behavioral model;
5   associating said behavioral model with a focus, said
6   focus having a reference to said behavioral model, and
7   having at least one directive; and
8   generating a test program according to state transi-
9   tions of said behavioral model and said directive of said
10   focus.

1   2. The method according to claim 1, wherein said di-
2   rective comprises a model independent directive.

1   3. The method according to claim 1, wherein said di-
2   rective comprises a model dependent directive, and a cov-
3   erage variable of said behavioral model is tagged by a
4   tag of said model dependent directive, said coverage
5   variable having allowable values.

1   4. The method according to claim 3, wherein said di-
2   rective further comprises a model independent directive.

1   5. The method according to claim 3, wherein said test
2   program references said coverage variable, and said step

3 of generating is performed until said coverage variable

4 has assumed each of said allowable values.

1   6. The method according to claim 5, wherein said cov-

2 erage variable comprises a plurality of coverage vari-

3 ables, and said step of generating is performed until a

4 cross product of said coverage variables has assumed all

5 possible values thereof.

1   7. The method according to claim 5, wherein said cov-

2 erage variable comprises a plurality of coverage vari-

3 ables, and said step of generating is performed until an

4 orthogonal array of said coverage variables has assumed

5 all possible values thereof.

1   8. The method according to claim 3, wherein said

2 model dependent directive comprises a plurality of model

3 dependent directives, and said coverage variable is

4 tagged by a plurality of tags of said model dependent di-

5 rectives.

1   9. The method according to claim 3, wherein said tag

2 is a number-of-tests-per-value tag.

1   10. The method according to claim 3, wherein said

2 model dependent directive is a mask-value directive.

1   11. The method according to claim 1, wherein said di-
2 rective comprises a plurality of directives that are com-
3 bined to define a directive expression, wherein said step
4 of generating is performed until said directive expres-
5 sion has a predetermined value.

1   12. The method according to claim 1, wherein said
2 step of modeling is performed by retrieving said behav-
3 ioral model from a model archive.

1   13. The method according to claim 1, wherein said
2 step of associating is performed by retrieving said focus
3 from a focus archive.

1   14. The method according to claim 13, further com-
2 prising the steps of comparing state variables of foci of
3 said focus archive with state variables of said behav-
4 ioral model; and
5   responsive to comparisons resulting from said step of
6 comparing revising selected ones of said foci.

1   15. A computer software product, comprising a com-
2 puter-readable medium in which computer program instruc-
3 tions are stored, which instructions, when read by a com-
4 puter, cause the computer to execute a method of test
5 generation for testing computer software, the method com-
6 prising the steps of:

7   accepting as a first input a behavioral model of a
8 software application, wherein said behavioral model com-
9 prises a finite state machine;

10   accepting as a second input a focus having a refer-
11 ence to said behavioral model, and having at least one
12 directive;

13   associating said behavioral model with said focus;
14 and

15   generating a test program according to state transi-
16 tions of said behavioral model and said directive of said
17 focus.

1   16. The computer software product according to
2 claim 15, wherein said directive comprises a model inde-
3 pendent directive.

1   17. The computer software product according to claim
2 15, wherein said directive comprises a model dependent
3 directive, and a coverage variable of said behavioral
4 model is tagged by a tag of said model dependent direc-
5 tive, said coverage variable having allowable values.

1   18. The computer software product according to
2 claim 17, wherein said directive further comprises a
3 model independent directive.

1   19. The computer software product according to
2 claim 17, wherein said test program references said cov-

3 erage variable, and said step of generating is performed

4 until said coverage variable has assumed each of said al-

5 lowable values.

1   20. The computer software product according to

2 claim 19, wherein said coverage variable comprises a plu-

3 rality of coverage variables, and said step of generating

4 is performed until a cross product of said coverage vari-

5 ables has assumed all possible values thereof.

1   21. The computer software product according to

2 claim 19, wherein said coverage variable comprises a plu-

3 rality of coverage variables, and said step of generating

4 is performed until an orthogonal array of said coverage

5 variables has assumed all possible values thereof.

1   22. The computer software product according to

2 claim 17, wherein said model dependent directive com-

3 prises a plurality of model dependent directives, and

4 said coverage variable is tagged by a plurality of tags

5 of said model dependent directives.

1   23. The computer software product according to

2 claim 17, wherein said tag is a number-of-tests-per-value

3 tag.

1   24. The computer software product according to
2 claim 17, wherein said model dependent directive is a
3 mask-value directive.

1   25. The computer software product according to
2 claim 15, wherein said directive comprises a plurality of
3 directives that are combined to define a directive ex-
4 pression, wherein said step of generating is performed
5 until said directive expression has a predetermined
6 value.

1   26. The computer software product according to
2 claim 15, wherein said step of modeling is performed by
3 retrieving said behavioral model from a model archive.

1   27. The computer software product according to
2 claim 15, wherein said step of associating is performed
3 by retrieving said focus from a focus archive.

1   28. The computer software product according to
2 claim 27, further comprising the steps of comparing state
3 variables of foci of said focus archive with state vari-
4 ables of said behavioral model; and
5   responsive to comparisons resulting from said step of
6 comparing revising selected ones of said foci.

1   29. A method of test generation for testing computer
2 software, comprising the steps of:

3   modeling a software application as a finite state ma-
4   chine to define a behavioral model;
5   associating said behavioral model with a focus, said
6   focus having a reference to said behavioral model, and
7   having at least one directive;
8   deriving an abstract test suite from said behavioral
9   model and said focus, wherein said abstract test suite
10   complies with a test constraint that is encoded in said
11   focus;
12   executing said abstract test suite in an execution
13   engine.

1   30. The method according to claim 29, wherein said
2   step of executing said abstract test suite comprises the
3   step of generating a test script from said abstract test
4   suite; wherein said test script is executed in said exe-
5   cution engine.

1   31. The method according to claim 29, wherein said
2   step of producing said abstract test suite is performed
3   with a testing interface.

1   32. The method according to claim 31, wherein said
2   testing   interface   comprises   an   abstract-to-concrete
3   translation table.

1   33. The method according to claim 29, wherein said
2 testing interface comprises a test driver, having an op-
3 erator interface, and further comprising the step of:
4   varying parameters of said test driver via said op-
5 erator interface in accordance with requirements of said
6 software application.

1   34. The method according to claim 29, wherein said
2 directive comprises a model independent directive.

1   35. The method according to claim 38 wherein said
2 coverage variable comprises a plurality of coverage vari-
3 ables, and said step of generating is performed until a
4 cross product of said coverage variables has assumed all
5 possible values thereof.

1   36. The method according to claim 38, wherein said
2 coverage variable comprises a plurality of coverage vari-
3 ables, and said step of generating is performed until an
4 orthogonal array of said coverage variables has assumed
5 all possible values thereof.

1   37. The method according to claim 29, wherein said
2 directive comprises a model dependent directive, and a
3 coverage variable of said behavioral model is tagged by a
4 tag of said model dependent directive, said coverage
5 variable having allowable values.

38. The method according to claim 37, wherein said abstract test suite references said coverage variable, and said step of generating is performed until said coverage variable has assumed each of said allowable values.

39. The method according to claim 37, wherein said directive further comprises a model independent directive.

40. The method according to claim 37, wherein said model dependent directive comprises a plurality of model dependent directives, and said coverage variable is tagged by a plurality of tags of said model dependent directives.

41. The method according to claim 37, wherein said tag is a number-of-tests-per-value tag.

42. The method according to claim 37, wherein said model dependent directive is a mask-value directive.

43. The method according to claim 29, wherein said directive comprises a plurality of directives that are combined to define a directive expression, wherein said step of generating is performed until said directive expression has a predetermined value.

Ver. 40388S4

1   44. The method according to claim 29, wherein said
2   step of modeling is performed by retrieving said behav-
3   ioral model from a model archive.

1   45. The method according to claim 29, wherein said
2   step of associating is performed by retrieving said focus
3   from a focus archive.

1   46. The method according to claim 29, further com-
2   prising the steps of comparing state variables of foci of
3   said focus archive with state variables of said behav-
4   ioral model; and
5   responsive to comparisons resulting from said step of
6   comparing revising selected ones of said foci.

1   47. A computer software product for testing computer
2   software, comprising a computer-readable medium in which
3   computer program instructions are stored, which instruc-
4   tions, when read by a computer, cause the computer to
5   perform the steps of:
6   associating a behavioral model of a software applica-
7   tion with a focus, said focus having a reference to said
8   behavioral model, and having at least one directive,
9   wherein said behavioral model models a finite state ma-
10   chine;
11   deriving an abstract test suite from said behavioral
12   model and said focus, wherein said abstract test suite

IL9-2000-0079

13 complies with a test constraint that is encoded in said
14 focus;
15   executing said abstract test suite in an execution
16 engine.

1   48. The computer software product according to
2   claim 47, wherein said step of executing said abstract
3   test suite comprises the step of generating a test script
4   from said abstract test suite; wherein said test script
5   is executed in said execution engine.

1   49. The computer software product according to
2   claim 47, wherein said step of producing said abstract
3   test suite is performed with a testing interface.

1   50. The computer software product according to
2   claim 49, wherein said testing interface comprises an ab-
3   stract-to-concrete translation table.

1   51. The computer software product according to
2   claim 49, wherein said testing interface comprises a test
3   driver, having an operator interface, and further com-
4   prising the step of:
5   varying parameters of said test driver via said op-
6   erator interface in accordance with requirements of said
7   software application.

1   52.   The   computer   software   product   according   to
2   claim 47, wherein said directive comprises a model inde-
3   pendent directive.

1   53.   The   computer   software   product   according   to
2   claim 29, wherein said directive comprises a model de-
3   pendent directive, and a coverage variable of said behav-
4   ioral model is tagged by a tag of said model dependent
5   directive, said coverage variable having allowable val-
6   ues.

1   54.   The computer   software   product   according   to
2   claim 53 wherein said coverage variable comprises a plu-
3   rality of coverage variables, and said step of generating
4   is performed until a cross product of said coverage vari-
5   ables has assumed all possible values thereof.

1   55.   The   computer   software   product   according   to
2   claim 53, wherein said coverage variable comprises a plu-
3   rality of coverage variables, and said step of generating
4   is performed until an orthogonal array of said coverage
5   variables has assumed all possible values thereof.

1   56.   The   computer   software   product   according   to
2   claim 53, wherein said abstract test suite references
3   said coverage variable, and said step of generating is
4   performed until said coverage variable has assumed each
5   of said allowable values.

1   57. The computer software product according to
2   claim 53, wherein said directive further comprises a
3   model independent directive.

1   58. The computer software product according to
2   claim 53, wherein said model dependent directive com-
3   prises a plurality of model dependent directives, and
4   said coverage variable is tagged by a plurality of tags
5   of said model dependent directives.

1   59. The computer software product according to
2   claim 37, wherein said tag is a number-of-tests-per-value
3   tag.

1   60. The computer software product according to
2   claim 37, wherein said model dependent directive is a
3   mask-value directive.

1   61. The computer software product according to
2   claim 47, wherein said directive comprises a plurality of
3   directives that are combined to define a directive ex-
4   pression, wherein said step of generating is performed
5   until said directive expression has a predetermined
6   value.

IL9-2000-0079

1   62.   The   computer   software   product   according   to
2   claim 47, wherein said step of modeling is performed by
3   retrieving said behavioral model from a model archive.

1   63.   The   computer   software   product   according   to
2   claim 47, wherein said step of associating is performed
3   by retrieving said focus from a focus archive.

1   64.   The   computer   software   product   according   to
2   claim 63, further comprising the steps of comparing state
3   variables of foci of said focus archive with state vari-
4   ables of said behavioral model; and
5   responsive to comparisons resulting from said step of
6   comparing revising selected ones of said foci.

1   65. A computer system for testing computer software,
2   comprising:
3   a user interface for creating a behavioral model of a
4   software application, said behavioral model representing
5   a finite state machine, wherein said user interface cre-
6   ates a focus, said focus having a reference to said be-
7   havioral model, and having at least one directive;
8   a compiler, for converting said behavioral model into
9   an intermediate encoding thereof;
10   a test generator, accepting said intermediate encod-
11 ing and said focus as input, and producing an abstract
12 test suite;

13   an execution engine for executing a test program of
14   said abstract test suite.

1   66. The system according to claim 65, wherein said
2   execution engine produces a suite execution trace.

1   67. The system according to claim 66, further com-
2   prising an analyzer which reads said suite execution
3   trace, wherein said execution engine accepts an output of
4   said analyzer.

1   68. The system according to claim 65, further com-
2   prising a visualizer for visualizing an output of said
3   execution engine.

1   69. The system according to claim 65, wherein said
2   execution engine further receives input from an applica-
3   tion model interface that is created by said user inter-
4   face.

1   70. The system according to claim 65, wherein said
2   directive comprises a model independent directive.

1   71. The system according to claim 65, wherein said
2   directive comprises a model dependent directive, and a
3   coverage variable of said behavioral model is tagged by a
4   tag of said model dependent directive, said coverage
5   variable having allowable values.

1   72. The system according to claim 71, wherein said
2 directive further comprises a model independent direc-
3 tive.

1   73. The system according to claim 71, wherein said
2 test program references said coverage variable, and said
3 test generator operates until said coverage variable has
4 assumed each of said allowable values.

1   74. The system according to claim 73, wherein said
2 coverage variable comprises a plurality of coverage vari-
3 ables, and said execution engine executes until a cross
4 product of said coverage variables has assumed all possi-
5 ble values thereof.

1   75. The system according to claim 73, wherein said
2 coverage variable comprises a plurality of coverage vari-
3 ables, and said execution engine executes until an or-
4 thogonal array of said coverage variables has assumed all
5 possible values thereof.

1   76. The system according to claim 71, wherein said
2 model dependent directive comprises a plurality of model
3 dependent directives, and said coverage variable is
4 tagged by a plurality of tags of said model dependent di-
5 rectives.

1   77. The system according to claim 71, wherein said
2 tag is a number-of-tests-per-value tag.

1   78. The system according to claim 71, wherein said
2 model dependent directive is a mask-value directive.

1   79. The system according to claim 65, wherein said
2 directive comprises a plurality of directives that are
3 combined to define a directive expression, wherein said
4 execution engine executes until said directive expression
5 has a predetermined value.

1   80. The system according to claim 65, further com-
2 prising a model archive that is accessed by said user in-
3 terface.

1   81. The system according to claim 65, further com-
2 prising a focus archive that is accessed by said user in-
3 terface.